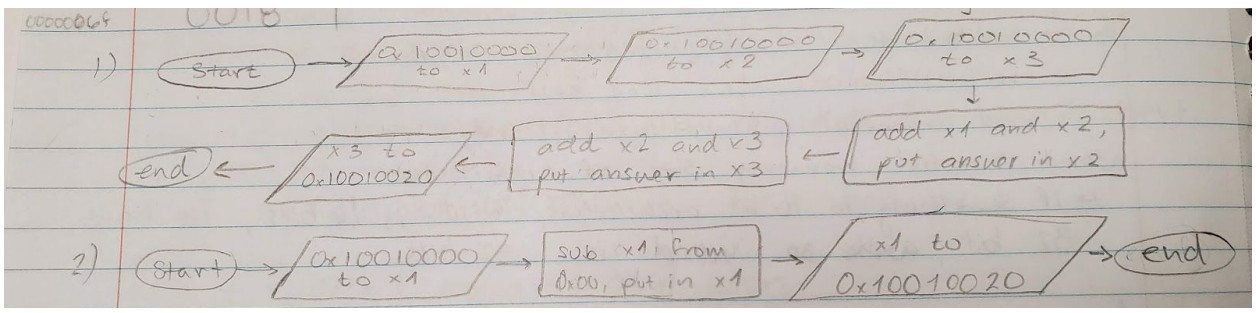
Zuzanna Dominik, Meha Sharma
CPE 233 Software Assignment 1
1/13/2020

**Behavior Description:**

1. Reads 3 inputs from address 0x10010000 ( by using 0x100100004 and 0x10010008 as well), and adds them together. Output the sum to address 0x10010020. Assumes the input values are 16-bit unsigned values.

2. Reads an input from address 0x10010000. Assumes the input is a 16-bit signed value in 2's complement (RC). Changes the sign of the input and outputs the result to address 0x10010020. The result is 16-bit signed value also in 2's complement.

**Flow Chart:**




**Verification By Simulation:**

1.

Edit | Execute

Text Segment

| Bkpt | Address | Code | Basic | Source |
|---|---|---|---|---|
| | 0x00400000 | 0x100100b7 | lui x1,0x00010010 | 1: lhu x1, 0x10010000 # load first 16-bit value into x1 |
| | 0x00400004 | 0x0000d083 | lhu x1,0x00000000(x1) | |
| | 0x00400008 | 0x10010137 | lui x2,0x00010010 | 2: lhu x2, 0x10010004 # load second 16-bit value into x2 |
| | 0x0040000c | 0x00415103 | lhu x2,0x00000004(x2) | |
| | 0x00400010 | 0x100101b7 | lui x3,0x00010010 | 3: lhu x3, 0x10010008 # load third 16-bit value into x3 |
| | 0x00400014 | 0x0081d183 | lhu x3,0x00000008(x3) | |
| | 0x00400018 | 0x00208133 | add x2,x1,x2 | 4: add x2, x1, x2 # add first and second value together, store in x2 |
| | 0x0040001c | 0x003101b3 | add x3,x2,x3 | 5: add x3, x2, x3 # add sum of last two and third value, store in x3 |
| | 0x00400020 | 0x100100b7 | lui x1,0x00010010 | 6: sh x3, 0x10010020, x1 # save the 16-bit sum in 0x10010020 |
| | 0x00400024 | 0x02309023 | sh x3,0x00000020(x1) | |
| | 0x00400028 | 0x0000006f | jal x0,0x00000000 | 7: end: j end # continuous loop to end program |

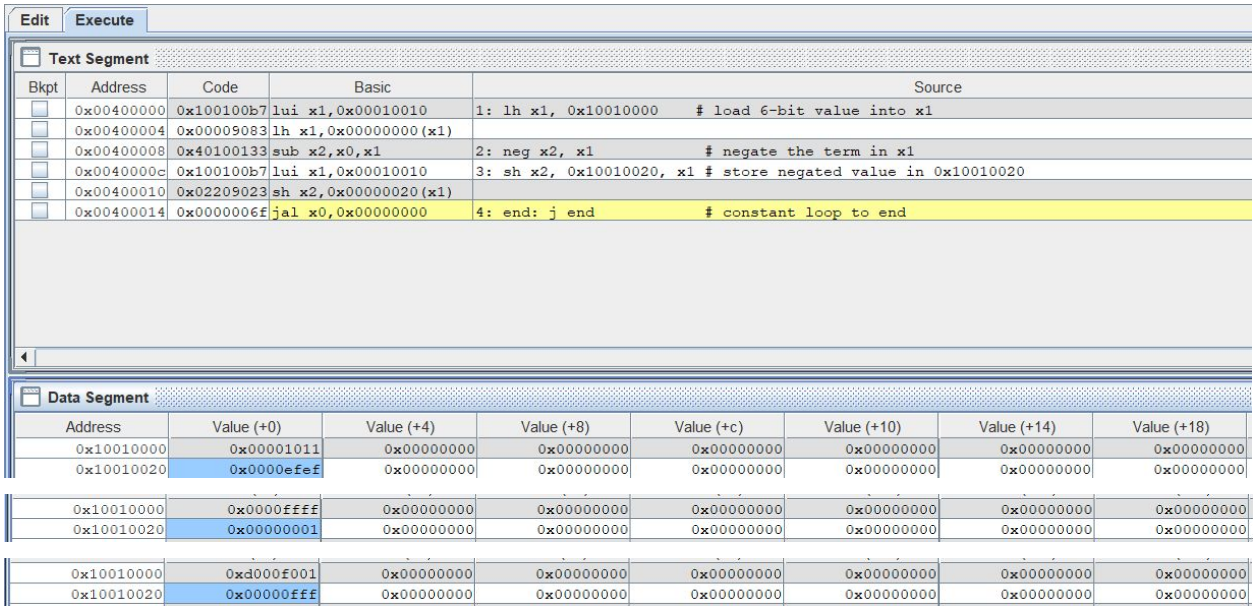
Data Segment

| Address | Value (+0) | Value (+4) | Value (+8) | Value (+c) | Value (+10) | Value (+14) | Value (+18) |
|---|---|---|---|---|---|---|---|
| 0x10010000 | 0x00000001 | 0x00000010 | 0x00000100 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010020 | 0x00000111 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x10010000 | 0xf000000a | 0x00000002 | 0x00000004 | 0xfff00000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010020 | 0x00000010 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x10010000 | 0x0000ffff | 0x0000ffff | 0x0000ffff | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010020 | 0x0000fffd | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

2.



Edit | Execute

Text Segment

| Bkpt | Address | Code | Basic | Source |
|---|---|---|---|---|
| | 0x00400000 | 0x100100b7 | lui x1,0x00010010 | 1: lh x1, 0x10010000 # load 6-bit value into x1 |
| | 0x00400004 | 0x00009083 | lh x1,0x00000000(x1) | |
| | 0x00400008 | 0x40100133 | sub x2,x0,x1 | 2: neg x2, x1 # negate the term in x1 |
| | 0x0040000c | 0x100100b7 | lui x1,0x00010010 | 3: sh x2, 0x10010020, x1 # store negated value in 0x10010020 |
| | 0x00400010 | 0x02209023 | sh x2,0x00000020(x1) | |
| | 0x00400014 | 0x0000006f | jal x0,0x00000000 | 4: end: j end # constant loop to end |

Data Segment

| Address | Value (+0) | Value (+4) | Value (+8) | Value (+c) | Value (+10) | Value (+14) | Value (+18) |
|---|---|---|---|---|---|---|---|
| 0x10010000 | 0x00001011 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010020 | 0x0000efef | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x10010000 | 0x0000ffff | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010020 | 0x00000001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x10010000 | 0xd000f001 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010020 | 0x00000fff | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

**Assembly Source Code:**

1. lhu x1, 0x10010000 # load first 16-bit value into x1
   lhu x2, 0x10010004 # load second 16-bit value into x2
   lhu x3, 0x10010008 # load third 16-bit value into x3
   add x2, x1, x2 # add first and second value together, store in x2
   add x3, x2, x3 # add sum of last two and third value, store in x3
   sh x3, 0x10010020, x1 # save the 16-bit sum in 0x10010020
   end: j end # continuous loop to end program

2. lh x1, 0x10010000 # load 16-bit value into x1
   neg x2, x1 # negate the term in x1
   sh x2, 0x10010020, x1 # store negated value in 0x10010020
   end: j end # constant loop to end